

Datenbanken und Datensicherheit

Prof. Dr. rer. nat. Peter Gerwinski

15. Januar 2025

Datenbanken und Datensicherheit

<https://gitlab.cvh-server.de/pgerwinski/dbs>

1 Einführung

2 Kurzeinführung Unix

3 Kurzeinführung TCP/IP

4 Relationale Datenbanken

...

4.11 Funktionen und Trigger

4.12 GUI-Zugriff

4.13 Datensicherheit bei Datenbanken

4.14 Sonstige Datenbanken

5 Kryptographie

5.1 Einführung

5.2 Symmetrische Verschlüsselung

5.3 Asymmetrische und hybride Verschlüsselung

5.4 Signaturen

5.5 Authentifizierung

5.6 Quantencomputer

6 Datensicherheit



Änderungen
vorbehalten

4 Relationale Datenbanken

4.11 Funktionen und Trigger

Funktionen:

- **PROCEDURE** entspricht einer **void**-Funktion in C.
- <https://www.postgresql.org/docs/15/sql-createprocedure.html>

Trigger:

- SQL-Standard: <https://www.sqltutorial.org/sql-triggers/>
- PostgreSQL: <https://www.postgresqltutorial.com/postgresql-triggers/creating-first-trigger-postgresql/>
- Anwendungsbeispiele:
 - Konsistenz
 - Protokollieren
 - Echtzeit-Aktualisierung bei DBMS-Migration

4 Relationale Datenbanken

4.12 GUI-Zugriff

- Anwendung nutzt DBMS-Client-Bibliothek
GUI-Programmierung: wie gewohnt
- Spezialfall: Web-Anwendung

Beispiel: Programmiersprache PHP

- Integration in HTML-Quelltext: `<?php ... ?>`
- Objekt zur Kommunikation mit Datenbanken: PDO

Literatur:

- <https://www.postgresqltutorial.com/postgresql-php/connect/>
- <https://www.phptutorial.net/php-pdo/pdo-connecting-to-postgresql/>

4 Relationale Datenbanken

4.13 Datensicherheit bei Datenbanken

- kein direkter Zugriff von außen auf die Datenbank
- Transportverschlüsselung
- feingranulare Benutzerrechte
- Software aktuell halten
- gegen SQL Injection: Prepared Statements

4 Relationale Datenbanken

4.13 Datensicherheit bei Datenbanken: SQL Injection

Problem:

- Ein böswilliger Benutzer gibt über eine Benutzerschnittstelle (z. B. ein Web-Interface) Daten ein (z. B. einen „Namen“), die Sonderzeichen enthalten, damit sie als SQL-Befehle ausgeführt werden.
- Beispiele: <https://xkcd.com/327/>, <https://www.heise.de/-10220617>

Suboptimale Lösung: Die Benutzerschnittstelle prüft die Daten auf Sonderzeichen und ersetzt diese durch geeignete Escape-Sequenzen

- ' durch '' ersetzen
- Funktion `CHR ()`
- Viele DBMS verstehen ein vorangestelltes \.

Bessere Lösung: *Prepared Statements*

- <https://www.postgresql.org/docs/current/sql-prepare.html>
- https://www.w3schools.com/php/php_mysql_prepared_statements.asp

4 Relationale Datenbanken

4.14 Sonstige Datenbanken

- Eingebettete Datenbanken:
Berkeley DB, SQLite
Software-Bibliothek, keine Client-Server-Struktur
- Nicht-relationale Datenbanken:
dokumentenorientierte Datenbanken, noSQL
Performanz wichtiger als Konsistenz
→ Applikationen stärker in Konsistenzprüfung eingebunden

5 Kryptographie

5.1 Einführung

Was ist Datensicherheit?

(CIA)

- Vertraulichkeit (confidentiality) → Verschlüsselung
- Integrität (integrity) → Konsistenzprüfungen, Prüfwerte, Signaturen
- Verfügbarkeit (availability) → Backups, Ausfallsicherheit

- Identifizierbarkeit (Authentizität, Nichtabstreitbarkeit, Zurechenbarkeit)
→ Passwörter, Signaturen
bzw.
- Anonymität (plausible Abstreitbarkeit, Nichtzurechenbarkeit)
→ Pseudonymisierung, Anonymisierung,
Verschlüsselung, Steganographie

5 Kryptographie

5.1 Einführung

Was ist Datensicherheit?

- (CIA)
- Vertraulichkeit (confidentiality) → Verschlüsselung
- Integrität (integrity) → Konsistenzprüfungen, Prüfwerte, Signaturen
- Verfügbarkeit (availability) → Backups, Ausfallsicherheit
- Identifizierbarkeit (Authentizität, Nichtabstreitbarkeit, Zurechenbarkeit)
→ Passwörter, Signaturen
bzw.
- Anonymität (plausible Abstreitbarkeit, Nichtzurechenbarkeit)
→ Pseudonymisierung, Anonymisierung,
Verschlüsselung, Steganographie → Kryptographie

5 Kryptographie

Kryptographie

- Verschlüsselung: symmetrisch, asymmetrisch, hybrid
- Hashes: Einwegfunktionen, Salt
- Signaturen, Zertifikate
- Schlüsselaustausch

5.2 Symmetrische Verschlüsselung

- Derselbe Schlüssel zum Ver- und Entschlüsseln
- Beispiele: Cäsar-Chiffre, monoalphabetische Substitution

5 Kryptographie

Kryptographie

- Verschlüsselung: symmetrisch, asymmetrisch, hybrid
- Hashes: Einwegfunktionen, Salt
- Signaturen, Zertifikate
- Schlüsselaustausch

5.2 Symmetrische Verschlüsselung

- Derselbe Schlüssel zum Ver- und Entschlüsseln
- Beispiele: Cäsar-Chiffre, monoalphabetische Substitution

5 Kryptographie

Kryptographie

- Verschlüsselung: symmetrisch, asymmetrisch, hybrid
- Hashes: Einwegfunktionen, Salt
- Signaturen, Zertifikate
- Schlüsselaustausch

5.2 Symmetrische Verschlüsselung

- Derselbe Schlüssel zum Ver- und Entschlüsseln
- Beispiele: Cäsar-Chiffre, monoalphabetische Substitution

Unsicher!



5 Kryptographie

Kryptographie

- Verschlüsselung: symmetrisch, asymmetrisch, hybrid
- Hashes: Einwegfunktionen, Salt
- Signaturen, Zertifikate
- Schlüsselaustausch

5.2 Symmetrische Verschlüsselung

- Derselbe Schlüssel zum Ver- und Entschlüsseln
- Beispiele: **Cäsar-Chiffre**, **monoalphabetische Substitution**, One Time Pad (beweisbar sicher)

Unsicher!



5 Kryptographie

Kryptographie

- Verschlüsselung: symmetrisch, asymmetrisch, hybrid
- Hashes: Einwegfunktionen, Salt
- Signaturen, Zertifikate
- Schlüsselaustausch

5.2 Symmetrische Verschlüsselung

- Derselbe Schlüssel zum Ver- und Entschlüsseln
- Beispiele: Cäsar-Chiffre, monoalphabetische Substitution, One Time Pad (beweisbar sicher), Pseudozufallszahlengenerator, Startwert als Schlüssel

Unsicher!



5 Kryptographie

Kryptographie

- Verschlüsselung: symmetrisch, asymmetrisch, hybrid
- Hashes: Einwegfunktionen, Salt
- Signaturen, Zertifikate
- Schlüsselaustausch

5.2 Symmetrische Verschlüsselung

Unsicher!

- Derselbe Schlüssel zum Ver- und Entschlüsseln
- Beispiele: Cäsar-Chiffre, monoalphabetische Substitution, One Time Pad (beweisbar sicher), *spezielle* Pseudozufallszahlengeneratoren, Startwert als Schlüssel: Enigma, DES, 3DES, RC4, IDEA, Blowfish, TwoFish, CAST, AES, ...

5 Kryptographie

Kryptographie

- Verschlüsselung: symmetrisch, asymmetrisch, hybrid
- Hashes: Einwegfunktionen, Salt
- Signaturen, Zertifikate
- Schlüsselaustausch

5.2 Symmetrische Verschlüsselung

Unsicher!

- Derselbe Schlüssel zum Ver- und Entschlüsseln
- Beispiele: Cäsar-Chiffre, monoalphabetische Substitution, One Time Pad (beweisbar sicher), *spezielle* Pseudozufallszahlengeneratoren, Startwert als Schlüssel: Enigma, DES, 3DES, RC4, IDEA, Blowfish, TwoFish, CAST, AES, ...
- Problem: Schlüsselaustausch

5 Kryptographie

Kryptographie

- Verschlüsselung: symmetrisch, asymmetrisch, hybrid
- Hashes: Einwegfunktionen, Salt
- Signaturen, Zertifikate
- Schlüsselaustausch

5.2 Symmetrische Verschlüsselung

Unsicher!

- Derselbe Schlüssel zum Ver- und Entschlüsseln
- Beispiele: **Cäsar-Chiffre**, **monoalphabetische Substitution**,
One Time Pad (beweisbar sicher),
spezielle Pseudozufallszahlengeneratoren, Startwert als Schlüssel:
Enigma, **DES**, **3DES**, **RC4**, IDEA, Blowfish, TwoFish, CAST, AES, ...
- Problem: Schlüsselaustausch
- Lösung: *asymmetrische Verschlüsselung*

5.3 Asymmetrische und hybride Verschlüsselung

- verschiedene Schlüssel zum Ver- und Entschlüsseln:
öffentlicher und privater Schlüssel
- Prinzip: mathematische Operation,
einfach durchzuführen, schwer rückgängig zu machen
- Beispiel: $N = p \cdot q$ – einfacher als Primfaktorzerlegung von $N \longrightarrow$ RSA
 $73 \cdot 97 = 7081$: geht notfalls noch im Kopf
Primfaktorzerlegung von 7081: mindestens schriftlich, besser mit Rechner
- Beispiel: $c = b^a$ – einfacher als $a = \log_b c \longrightarrow$ Diffie-Hellman, ElGamal
 $7^5 = 16807$: geht notfalls noch im Kopf
 $\log_7 16807$: mindestens schriftlich, besser mit Rechner

→ Details: Algorithmen und Datenstrukturen

5.3 Asymmetrische und hybride Verschlüsselung

- verschiedene Schlüssel zum Ver- und Entschlüsseln:
öffentlicher und privater Schlüssel
- Prinzip: mathematische Operation,
einfach durchzuführen, schwer rückgängig zu machen
- Beispiel: $N = p \cdot q$ – einfacher als Primfaktorzerlegung von $N \longrightarrow$ RSA
 $73 \cdot 97 = 7081$: geht notfalls noch im Kopf
Primfaktorzerlegung von 7081: mindestens schriftlich, besser mit Rechner
- Beispiel: $c = b^a$ – einfacher als $a = \log_b c \longrightarrow$ Diffie-Hellman, ElGamal
 $7^5 = 16807$: geht notfalls noch im Kopf
 $\log_7 16807$: mindestens schriftlich, besser mit Rechner

→ Details: Algorithmen und Datenstrukturen

- Nachteil: wesentlich aufwendiger und daher langsamer
als symmetrische Verschlüsselung

→ *hybride Verschlüsselung*: nur Schlüsselaustausch asymmetrisch,
eigentliche Verschlüsselung symmetrisch

5.4 Signaturen

- *kryptographische Hash-Funktion*: leicht auszurechnen, schwer zu manipulieren
- asymmetrisch: *Signatur*
Hash-Wert mit privatem Schlüssel verschlüsseln,
mit öffentlichem Schlüssel entschlüsseln
- symmetrisch: *Message Authentication Code* (MAC)
z. B. Hash-Wert über Nachricht + geheimer Schlüssel

5.4 Signaturen

- *kryptographische Hash-Funktion*: leicht auszurechnen, schwer zu manipulieren
- asymmetrisch: *Signatur*
Hash-Wert mit privatem Schlüssel verschlüsseln,
mit öffentlichem Schlüssel entschlüsseln
- symmetrisch: *Message Authentication Code* (MAC)
z. B. Hash-Wert über Nachricht + geheimer Schlüssel

Angriffsmöglichkeit: *Man-in-the-middle*-Angriff

Beim Schlüsselaustausch anderen Schlüssel unterschieben

→ Sorgfalt beim Schlüsselaustausch

5.4 Signaturen

- *kryptographische Hash-Funktion*: leicht auszurechnen, schwer zu manipulieren
- asymmetrisch: *Signatur*
Hash-Wert mit privatem Schlüssel verschlüsseln,
mit öffentlichem Schlüssel entschlüsseln
- symmetrisch: *Message Authentication Code* (MAC)
z. B. Hash-Wert über Nachricht + geheimer Schlüssel

Angriffsmöglichkeit: *Man-in-the-middle*-Angriff

Beim Schlüsselaustausch anderen Schlüssel unterschieben

→ Sorgfalt beim Schlüsselaustausch

Praxis-Beispiele

- SSH
- HTTPS
- OpenPGP → Praktikumsversuch 3

5.5 Authentifizierung

- Zugangsdaten:
Benutzername, Passwort
- Problem:
Zugangsdaten mitlesen
- Lösung:
verschlüsselte Verbindung
- Problem:
Zugangsdaten speichern
- Lösung:
Hash-Wert statt Passwort speichern
- Problem:
gleiche Passwörter identifizierbar
- Lösung:
Salt

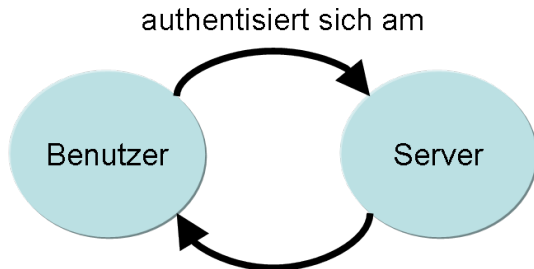


Bild: <https://de.wikipedia.org/wiki/Datei:Authentisieren-authentifizieren.png>

Hash-Algorithmen:

CRC → *kein* Hash-Algorithmus!

DES – unsicher (Unix traditionell)

MD5 – unsicher

SHA – Stand der Technik

weitere Algorithmen:

siehe *man 5 crypt*

5.5 Authentifizierung

- Zugangsdaten:
Benutzername, Passwort
- Problem:
Zugangsdaten mitlesen
- Lösung:
verschlüsselte Verbindung
- Problem:
Zugangsdaten speichern
- Lösung:
Challenge-Response-Authentifizierung
Beispiel: HTTP Digest

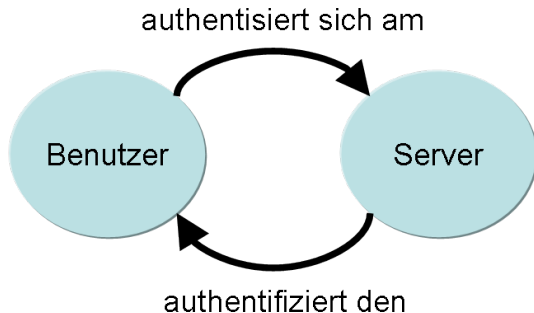


Bild: <https://de.wikipedia.org/wiki/Datei:Authentisieren-authentifizieren.png>

- gemeinsamer geheimer Schlüssel (Passwort)
- Server schickt *Nonce* an Benutzer
- Benutzer schickt Hash über [Passwort + Nonce] an Server
- Server berechnet denselben Hash → Authentifizierung erfolgreich
- Nonce (= „number used once“) nur einmal verwenden!

5.6 Quantencomputer

Analogie: Spaghetti-Sort

https://en.wikipedia.org/wiki/Spaghetti_sort

- Prinzip: 2^n Berechnungen gleichzeitig
(n = Registerbreite)

→ Klassisch schwierige Probleme werden einfacher.

- Beispiel: Primfaktorzerlegung: $\mathcal{O}(n^3)$ statt $\mathcal{O}(2^{\sqrt{n \log n}})$
- Problem für asymmetrische Verschlüsselungsalgorithmen,
z. B. RSA (Primfaktorzerlegung), ElGamal (diskreter Logarithmus)
- weniger problematisch für symmetrische Verschlüsselungsalgorithmen

→ *Post-Quanten-Kryptographie*

Beispiel: McEliece-Kryptosystem

→ *Forward Secrecy*

Kompromittierung betrifft nur zukünftige Kommunikation,
nicht bereits vergangene. Beispiel:

- RSA nur für Authentifizierung
- Austausch eines Sitzungsschlüssels via Diffie-Hellman
- Kommunikation über Sitzungsschlüssel (symmetrisch)
- Sitzungsschlüssel nur einmal verwenden!

6 Datensicherheit

6.1 Prinzipien: Wie erreichen wir Datensicherheit?

Komplexitätsreduzierung

- Teilsysteme gegeneinander abgrenzen, Abhängigkeiten reduzieren
- fehlertolerantes Design

Realistische Zeitplanung

- ... anstelle von:
Probleme lange vor sich her schieben, dann alles auf einmal einfordern („gleichzeitig zu langsam und zu schnell“ – Martin Tschirsich, 2025)
- Problem: wirtschaftliche oder politische Zeitvorgaben
- Ziele nicht nachträglich ändern, sondern frühzeitig festlegen

Verantwortungen klären und ausüben

- Beteiligten, die technisch Ahnung haben, Zeit, Freiraum und Autorität geben

Transparenz

- Prozesse dokumentieren, auch Entwicklungsprozesse, offene Quelltexte

Positivbeispiel: Corona-Warn-App

6 Datensicherheit

6.2 Netzwerksicherheit

- Firewall: nur bestimmte IP-Adressen / Ports / Inhalte zulassen
- VPN: verschlüsselte Verbindung von Netzwerken über ansonsten unsichere Verbindung (Internet)
- *Intrusion Detection System*

Anonymität

- Beispiel: Tor – Zwiebel-Routing
 - Tor-Browser
 - Tails
- Beispiel: Corona-Warn-App

Cross-Site-Scripting verhindern

6 Datensicherheit

6.2 Netzwerksicherheit

Die menschliche Komponente

- Bequemlichkeit
- Social Engineering
 - *Phishing*
 - KI-Sprachmodelle

*Es gibt für jedes menschliche Problem
immer eine wohl bekannte Lösung -
sauber, einleuchtend, und falsch.*

Henry Louis Mencken, The Divine Afflatus, 1917
<https://de.wikiquote.org/wiki/Lösung>

6 Datensicherheit

6.3 Verfügbarkeit

Wann wird wirklich auf den Datentäger geschrieben?

- DBMS: Persistenz-Einstellungen
- *Write Ahead Log (WAL)*
Journaling-Dateisysteme
- *CAP-Theorem*

Daten sicher aufbewahren

- Backup
- RAID

Hochverfügbarkeit

- allgemein: *High-Availability-Cluster*
- speziell: Datenbank-Cluster: Replikation über mehrere Server

6 Datensicherheit

6.4 Datenschutz

- Schutz vor mißbräuchlicher Datenverarbeitung
- informationelle Selbstbestimmung
- Persönlichkeitsrecht
- Privatsphäre

Datenmißbrauch ermöglicht hohe Gewinne

- viel Interesse an persönlichen Daten
- „Datenschutz hemmt ~~den Fortschritt!~~“ die persönliche Bereicherung

- DSGVO

Datenbanken und Datensicherheit

<https://gitlab.cvh-server.de/pgerwinski/dbs>

1 Einführung

2 Kurzeinführung Unix

3 Kurzeinführung TCP/IP

4 Relationale Datenbanken

5 Kryptographie

5.1 Einführung

5.2 Symmetrische Verschlüsselung

5.3 Asymmetrische und hybride Verschlüsselung

5.4 Signaturen

5.5 Authentifizierung

5.6 Quantencomputer

6 Datensicherheit

6.1 Prinzipien: Wie erreichen wir Datensicherheit?

6.2 Netzwerksicherheit

6.3 Verfügbarkeit

6.4 Datenschutz

***Vielen Dank für Ihre Aufmerksamkeit
und viel Erfolg bei den Prüfungen!***